Documentation

This app facilitates conducting the staircase method to elicit individual level risk preferences, as proposed by Falk et al. (2016a, 2016b), as an *oTree* application (Chen et al., 2016) in different variants and parameterizations by simply altering the documented variables in `config.py`.

> Please note that, starting with oTree version 1.2.x, the template `Base.html` (in _templates/global) has been renamed to `Page.html`. If you wish to stick to an older version, please replace `{% extends "global/Page.html" %}` by `{% extends "global/Base.html" %}` in all HTML-templates in the app folder.

Installation

To install the app to your local *oTree* directory, copy the folder “icl” to your *oTree* Django project and extent the session configurations in your `settings.py` at the root of the *oTree* directory by something like

```
SESSION_CONFIG = [
   ...
   {
      'name': 'icl',
      'display_name': "Staircase Risk Elicitation",
      'num_demo_participants': 1,
      'app_sequence': ['icl'],
   },
   ...
]
```

Please note that global settings as `REAL_WORLD_CURRENCY_CODE`, `USE_POINTS`, as well as `SESSION_CONFIG_DEFAULTS` (including `participation_fee` and `real_world_currency_per_point`) are – as for all *oTree* apps – specified in *oTree*’s `settings.py` rather than the application itself but do affect the display of currency figures as well as the calculations of payoffs and amounts to pay.

Setup

To set up the task, the only thing to be done is to alter pre-defined variables in the file `config.py` at the root of the app's directory. Any combination of the variables described below is operable. By that means, several different variations of the task are easily implemented. The following variables can be specified:

`num_choices` (integer field):

Number (`n`) of subsequent choices with `i = 1, 2, …, n` in the "tree" (see Fig. 1). Note that `num_choices = x` implicitly defines a multiple price list with `2^x` binary choices, i.e. `2^x` possible switching points to classify individual level risk preferences. Thus, for instance, `num_choices = 5` implies only 5 decisions to be made but 32 consistent choices in the underlying multiple choice list.

`lottery_hi` and **`lottery_lo`** (decimal/currency fields):

`lottery_hi` and `lottery_lo` determine the "high" and the "low" payoff of the lottery ("Option A"). The lottery payoffs remain constant for all `num_choices` decisions. (Please note that the currency of payoffs displayed to subjects is determined by the global settings of *oTree* in `settings.py`.)

`probabilty` (integer/decimal field):

`probability` determines the likelihood of outcome "high" as a percentage number for the lottery in all decisions to be made, i.e. `probability = x` implies an `x`% chance that the prospect pays `lottery_hi` and a `(1-x)`% chance of yielding `lottery_lo`. The probability of lottery payoffs is held constant for all `num_choices` decisions.

`sure_payoff` (decimal/currency field):

`sure_payoff` defines the (initial) sure payoff, i.e. the certain payment ("Option B") in the first choice (`i = 1`). The sure payoffs for subsequent choices are determined by `delta` (see below), relative to `sure_payoff`. (Please note that the currency of payoffs displayed to subjects is determined by the global settings of *oTree* in `settings.py`.)

`delta` (decimal/currency field):

`delta` defines the (initial) increase or decrease, respectively, in the sure payoff in "Option B" while changes in sure payoffs of subsequent choices are defined as a function of `delta` and `num_choices`. Generally, for choice `i = 1, 2, ... num_choices`, the sure payment in "Option B" is defined by `sure_payoff(i) = sure_payoff(i-1) + delta / 2^(i-1)` if choice `(i-1) = "A"` and `sure_payoff(i) = sure_payoff(i-1)- delta / 2^(i-1)` if choice `(i-1) = "B"`. Thus, if a subject chooses "A" ("B"), the sure payoff in choice i increases (decreases) by half of the increase/decrease of the choice `(i-1)`, i.e. `delta(i-1)`. For example, if `sure_payoff = x` `delta = y`, the certain payment offered equals `x +/- y/2` in choice 2, `x +/- y/2 +/- y/4` in choice 3, etc. (see Fig. 1 for more details).

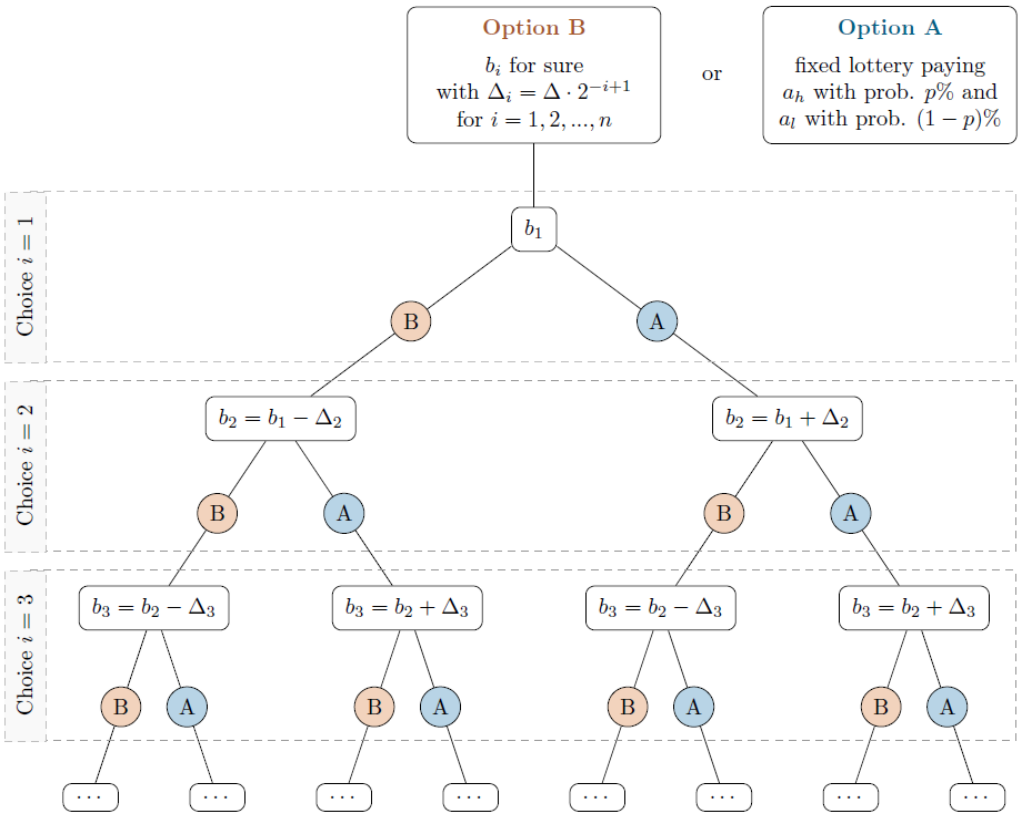

Figure 1. Staircase procedure. The lottery ("Option A") remains constant for all `num_choices` choices. "Option B" – a sure payoff, i.e. with a probability of 1 – changes in each iteration step, depending on the decision ("A" or "B") in the previous choice. "Option B" in choice *i = 1* (b_1) is determined by `sure_payoff`. Depending on the decision in *i = 1*, the sure payoff in choice *i = 2* (b_2) increases or decreases by `delta` (Δ). Generally, the sure payoff b_i in round *i = 1, 2, …, n* is defined by $b_i = b_i\text{-}1 + \Delta_i$ with $\Delta_i = \Delta \cdot 2^{-i+1}$, i.e. the sure payoff increases or decreases by half of the previous choice's delta each round.

`indifference` (boolean field):

If `indifference = False`, subjects are asked to reveal their preference for either "Option A" or "Option B"; if `indifference = True`, an additional option (“Indifferent”) is available to indicate indifference between Option "A" and "B", inspired by choice list procedures used, for instance, by Hey/Orme (1994) or Dimmock et al (2016). Note that the iteration procedure stops whenever a subject opts for the choice “Indifferent” as indifference is already reached, i.e., all subsequent choices are automatically skipped. If the "Indifferent" choice is drawn for payment, it is randomly determined whether "A" or "B" constitutes the payoff-relevant alternative.

`buttons` (boolean field):

`buttons = True` allows to render buttons instead of the default radio buttons for all options, i.e. for “Option A”, “Option B”, and – if `indifference = True` – for “Indifferent”. If `buttons = False`, radio buttons will be displayed. That is, if `buttons = True`, subjects only click a single button than rather choosing a radio button and clicking on `"Next"`, facilitating (and accelerating) the input of choices but implying that decisions cannot be modified before submission.

`progress_bar` (boolean field):

A progress bar, optionally, allows for graphical highlighting of the subject’s advance within the task, in terms of how many decision have already been completed. If `progress_bar = True` a progress bar is rendered; if `progress_bar = False`, no information with respect to the advance within the task is displayed. Furthermore, information in terms of "page *x* out of `num_choices`" (with *x* denoting the current decision) is provided.

`instructions` (boolean field):

If `instructions = True`, a separate template `Instructions.html` is rendered prior to the task in round one. If `instructions = False`, the task starts immediately (e.g. in case of printed instructions). Please note that the instructions included serve only exemplary purposes and need to be adjusted to your settings in `config.py`.

`results` (boolean field):

Determines whether a results page summarizing game outcome is rendered or not. If `results = True`, a separate view `Results.html` containing all relevant information (i.e. the randomly picked lottery to be payed, the randomly determined outcome to pay, and the payoff) is rendered. If `results = False`, no separate page is displayed.

References

Chen, D. L., Schonger, M., Wickens, C., 2016. “oTree – an open-source platform for laboratory, online and field experiments”. *Journal of Behavioral and Experimental Finance* 9, 88–97.

Dimmock, S. G., Kouwenberg, R., Mitchell, O. S., Peijnenburg, K., 2016. “Ambiguity aversion and household portfolio choice puzzles: Empirical evidence”. *Journal of Financial Economics* 119, 559–577.

Falk, A., Becker, A., Dohmen, T., Enke, T., Huffman, D., Sunde, U., 2016a. “The nature and predictive power of preferences: Global evidence”. *Human Capital and Economic Opportunity Global Working Group, Working Paper 2016-004*.

Falk, A., Becker, A., Dohmen, T., Huffman, D., Sunde, U., 2016b. “The preference survey module: A validated instrument for measuring risk time, and social preferences”. *Working Paper*.

Hey, J. D., Orme, C., 1994. “Investigating generalizations of expected utility theory using experimental data”. *Econometrica* 62 (6), 1291–1326.